DEEP TRANCE

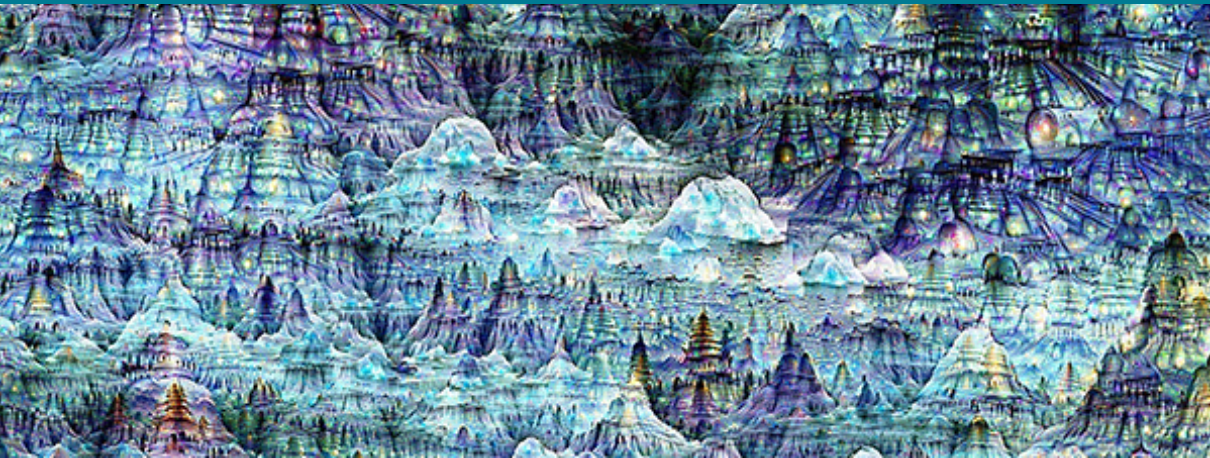
CSE204 Machine learning

Remy Seassau, Elouan Gros, Nazila Sharifi Amina

November 27, 2022

TABLE OF CONTENTS

INTRODUCTION: AN OVERVIEW OF GOOGLE'S DEEP DREAM



GOOGLE DEEP DREAM

- Instantiated by Google research in 2014
- Goal was to better understand neural networks

GOOGLE DEEP DREAM

- Instantiated by Google research in 2014
- Goal was to better understand neural networks
- A nice side effect was some cool art

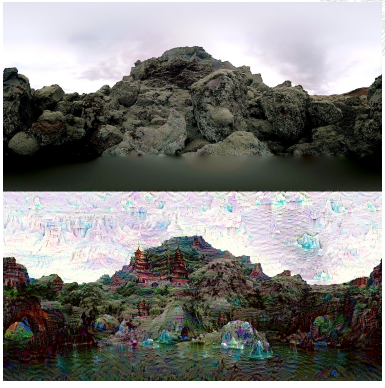


Figure: Showcase of Deep Mind on Google's AI Blog

FROM DREAM TO TRANCE

- We really were impressed by the artistic capabilities of Deep Mind
- We wondered if Google's techniques were also applicable to music
- We thus named our project Deep Trance and started searching

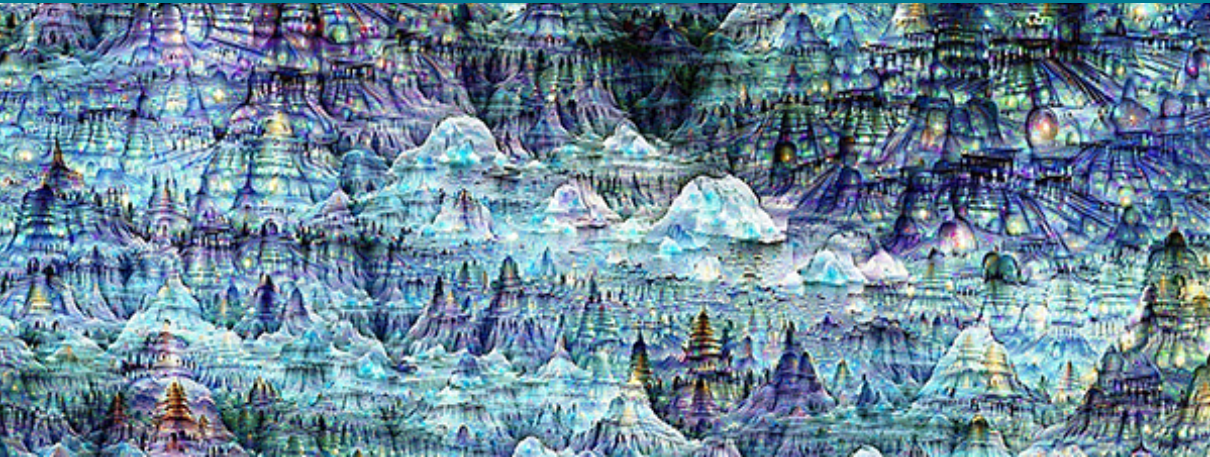
A SPECIFIC TYPE OF VISUALS

One of the more interesting features of the google deep dream experience was the different types of visualisations possible. Indeed, different models trained to recognize different features would manifest their specificities by making certain patterns appear withing images.

A SPECIFIC TYPE OF VISUALS

Figure: Deep Dream Visualisation on the Mona Lisa

A THEORETIC APPROACH TO DEEP DREAM



INNER WORKINGS OF DEEP DREAM

Deep dream's functionalities rest on a pretrained DNN image classifier.
The actual "dream" however, is nontrivial:

- The goal of the DNN is to optimize a model to recognize a label from a given input
- The goal of Deep Dream was to show the inner workings of an image classifier neural network
- Deep Dream can be thought of as a reversing the machine learning process

ON DREAMING

The "dreaming" thus goes along these lines:

1. An image is fed into the model
2. (optional) the model is interrupted at a given layer
3. The error is computed on the layer the computation was stopped on
4. The error is propagated backwards
5. We apply a gradient ascent algorithm to the image

This process can be repeated any number of times, emphasizing patterns

DEEP TRANCE

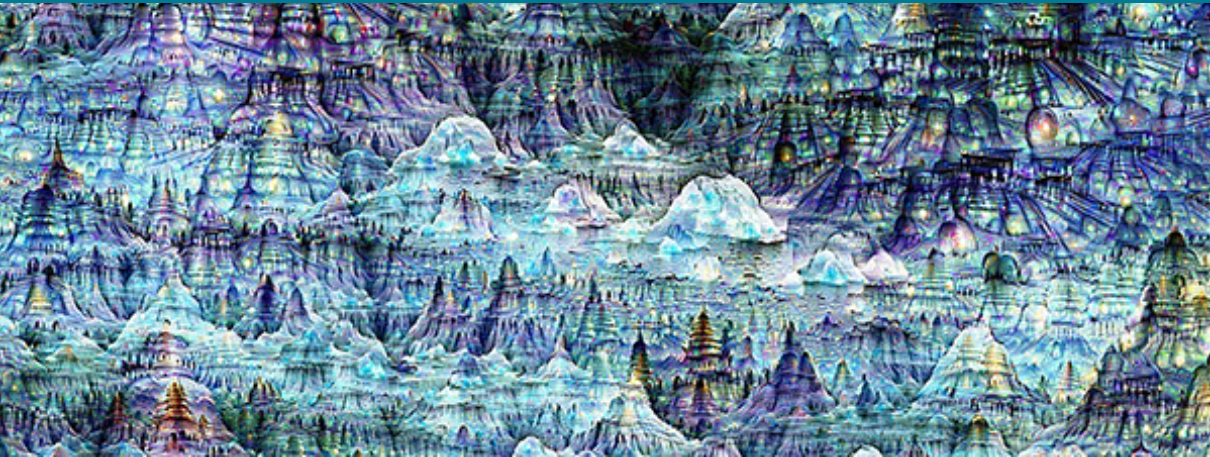
From Deep Dream's specification and our class we saw the following parallels:

- Like flattened images, music can be interpreted as a vector
- Like Deep Dream, we had to start by fashioning a classifying model
- We noticed that there is a bijection between songs (sound) and their spectrograms (images)

OUR PLAN

1. Train a Neural Network to recognize a song's genre from it's samples
2. Feed a song into the model
3. Compute the error
4. Change the samples of the song
5. repeat until the error is small enough

IMPLEMENTATION PROCESS



DATASET

The data set we decided to use is the one provided by the Free Music Archive.
It is free, hence we were able to use it for our project.

Moreover, The whole data set is already labeled, saving us precious time.
Finally, the data set and its usage is well documented, once again saving time.

DATASET

After downloading our dataset, we had:

DATASET

After downloading our dataset, we had:

- 8000 individual 30 second mp3 tracks

DATASET

After downloading our dataset, we had:

- 8000 individual 30 second mp3 tracks
- A csv file containing the features of 106574 tracks (including the 8000 tracks we downloaded)

DATASET

After downloading our dataset, we had:

- 8000 individual 30 second mp3 tracks
- A csv file containing the features of 106574 tracks (including the 8000 tracks we downloaded)

DATASET

After downloading our dataset, we had:

- 8000 individual 30 second mp3 tracks
- A csv file containing the features of 106574 tracks (including the 8000 tracks we downloaded)

These features included, Chroma, Tonnetz, Mel Frequency Cepstral Coefficient (MFCC), Spectral centroid, Spectral bandwidth, Spectral contrast,...

DATASET

After downloading our dataset, we had:

- 8000 individual 30 second mp3 tracks
- A csv file containing the features of 106574 tracks (including the 8000 tracks we downloaded)

These features included, Chroma, Tonnetz, Mel Frequency Cepstral Coefficient (MFCC), Spectral centroid, Spectral bandwidth, Spectral contrast,...

Important notice: It is difficult to change these features in a song on purpose

DATASET CLEANING

- Removed all the sub-genres by replacing them by the major genre they belonged to
- Removed the genres we couldn't identify ourselves
- Normalized all the songs to the same length

A LOOK AT OUR DATASET

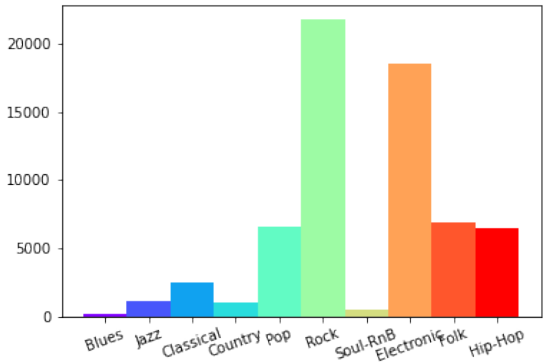


Figure: The distribution of our data by genre

A LOOK AT OUR DATASET

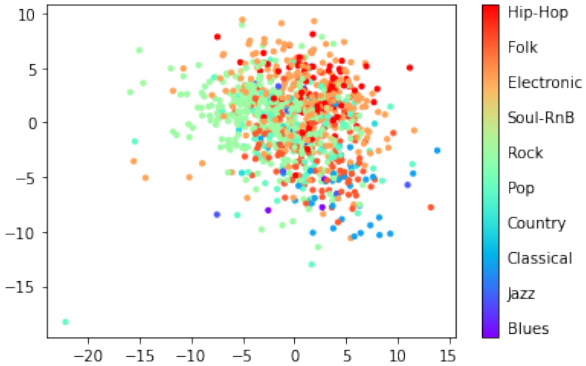


Figure: Results of an autoencoder on our genres

SAMPLE ANALYSIS

Our first attempt was to analyse the song samples by analysing their sample array.

However, we encountered a few problems:

- Even though we used the "small" data set, we were limited by available hardware (loading songs would take quite some time)
- We tried using only 10% of the small data set, which alleviate our loading problem
- The sample arrays were just too large (millions of points) for weight matrices to be built

We thus moved on to a second attempt.

KEY FEATURE ANALYSIS

Next we decided to use the data features csv as input to predict the genre

This came with some advantages

- Loading the dataset was very quick
- The dataset was much larger
- Very minimal preprocessing was needed
- We managed to get 60% accuracy using a standard model with 5 hidden layers and dropouts between them

However as we mentioned before, it was impossible to continue with the second phase of the project (the "dreaming") with these features as we could not feasibly correlate them so changes to a song

SPECTROGRAM ANALYSIS

Our third and final attempt at a classifier was built on our song's spectrograms:

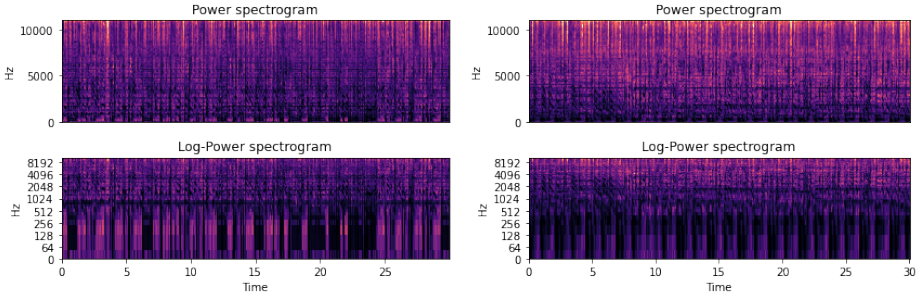


Figure: Two of our songs' spectrograms

SPECTROGRAM ANALYSIS

Using spectrograms proved somewhat easy because it reduces the problem down to image classification. However, we were reintroduced to the problem of loading times and small datasets.

We ended up working on datasets of about 1000 songs, which would take 45min to load and transform into spectrograms.

We used a convolution neural network and ended with a 90% accuracy on our training data and 20% on the testing data (clearly overfitting).

We believe this is due to the small size of our dataset and the large variance between the spectrograms of different songs (it would require a lot more data for common points between genres to appear).

DREAMING

We trained a binary classification model. We were planning on using it to Dream from various sounds. The results we expected were the following

- Sound distortion
- No audible change or a down-sampling
- Music altered in a "trippy" way like Deep Dream images
- Original song but shifted to a different song

However, the model didn't have enough data to reliably predict anything from the spectrograms and ended up predicting the same label no matter the input.

DREAMING

After the failure of our binary classification model, we decided to change to a multi-classifier in order to force our model to try and detect something.

This would prove half successful as the classifier was able to correctly identify the data it had been trained on but remained no better than a random guess for the rest.

We thus tried to run the dreaming algorithm on this model anyway, not sure of what to expect...

SOURCES

- Deep Dream blog post
- Deep Dream GitHub page
- Audio Data Analysis in Python, Nagesh Singh Chauhan
- Music Genre Classification Using CNN, Arsh Chowdhry